GPS Receiver Test

Local Application

Robert Goodwin

Wed, Aug 4, 2010

Local application GPST was written in 2002 to read time-of-day from a GPS receiver, model VME-SYNCCLOCK32, in preparation for support of accurate time-of-day for MiniBooNE. This note attempts to describe this LA.

Purpose

Accurate time-of-day information is needed today not only for MiniBooNE, but also for GETS32 protocol time stamp support. GPS receiver hardware allows comparison of the accuracy of this time-of-day support, delivered via the TIME local application, with independent reference. Every 60 seconds, TIME delivers via multicast the GMT time-of-day of the most recent 15 Hz clock event 0C. Until the next minute, each front end maintains its own copy, called GMT0C.

Parameters

The parameter layout for GPST is as follows:

ENABLE B	2	Usual LA enable Bit#
EVENT	2	Event# of interest, say, 8F
GPS	4	GPS receiver register base address, say, 0x4F000000
THRESH	2	Deviation threshold for logging, in μs
LOG CNTR	2	Total count of logged entries
DIFF	2	Difference between event time w/ GPS receiver and via GMT0C
DIFF MIN	2	Difference minimum
DIFF MAX	2	Difference maximum

For a given EVENT, compare the time of the event occurrence as derived from GMT0C with the time of the event obtained from the GPS receiver. (Given GMT0C and the microsecond counter time of any event's most recent occurrence, the GMT time can be computed for any event.

Event 8F is special. Rather than comparing the 8F event time as derived from the GMT0C global support, the comparison is made between the time from the GPS hardware and the calendar second, as every 8F event is expected to occur on a calendar second mark. (Another GPS receiver delivers a pulse that is inserted into the clock signal as event 8F.) In this way, GPST can be used, and *is* used, to measure how accurately the 8F events occur on calendar second marks.

There is an internal log maintained by GPST in an allocated memory block. It logs occurrences of deviations of the specified event from the expected time-of-day derived from GMT0C. Such deviations are expected to be small, but any deviation that is larger (in absolute value) than THRESH is logged. The additional parameters relate to this difference. To find the internal log, the address of its allocated block is at offset 0x0C from the start of GPST static memory block. After a 16-byte header, there is room for 63 entries in the circular buffer, each of size 16 bytes.

Operation

Each 15 Hz cycle, watch for the occurrence of the selected event via function HaveEvt, which returns true if it occurred at a time within the last cycle. Use the EventGMT function to obtain the calendar time of this event occurrence. Get the current time-of-day from the GPS receiver. Compare these two times, correcting for how long ago the event occurred, as obtained from the function GetEvtT. Log the difference if its absolute value exceeds the threshold value. (For the selected event 8F, the difference is defined as the deviation from a second mark.)

Details

The function `GetGPS` reads the current time from the GPS receiver. It first reads the long word at address `GPS+0x18`, then the long word at `GPS+0x1C`. The former holds the BCD for the current seconds (2 digits) and microseconds (6 digits). It converts the seconds and microseconds into binary. It is the microseconds that is used for comparison purposes; the two times to be compared are not expected to be too far apart.

The function `GetExtGPS` reads latched external time from the GPS receiver. This time is produced by the GPS receiver when it sees an external trigger, such as a timer pulse derived from decoding the `8F` clock event. This time was expected to be how we monitor the `8F` events, but it may be difficult because the GPS receiver may be busy just when we want to read the result. Because we run asynchronous with calendar time, it may be busy working out this time at any moment. This is why we use the current time approach.

Inside `GetExtGPS`, the logic is similar to `GetGPS`, but the long words are addressed at `GPS+0x28` and `GPS+0x2C`. In addition, one byte, out of 33, is read from the satellite status based at dual port ram address `0xA3`. This data, at `gpsData`, is not used, but merely allows one to examine it as a diagnostic. (What these data mean is a mystery to this writer.) Access to each dual port ram byte is done via the function `GetDPRam`, which writes the dual port ram address desired, waits up to 100 μs for it to be ready, then returns the byte accessed.